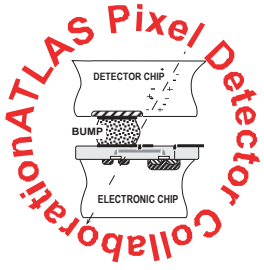

MCC-DSM Specifications

MCC Design Group

Receiver

Issue:	DRAFT
Revision:	0.1
Reference:	ATLAS ID-xx
Created:	30 September 2002
Last modified:	7 October 2002 03:18
Edited By:	R. Beccherle and G. Darbo
	ATLAS / Pixel Collaboration

Abstract

This document describes the Receiver of the *MCC-DSM*.

There are 16 identical copies of the Receiver inside the *MCC*.

Table Of Contents

List of Figures

List of Tables

1 Receiver

This chapter will describe the Receiver block of the *MCC*.

The architecture that we have chosen for the read-out of the pixel detector is a two level hierarchical structure that starts from the *FE* chips, and goes to the *MCC*. The interconnections between the FE chips and the MCC are unidirectional links. Being 16 *FE* chips hosted on a module there are 16 identical Receiver blocks inside the *MCC*.

The transmission protocol is “data push”: the *FE* chips send the hit coordinates as soon as they are ready in the end-of-column buffers. Since the chosen architecture is data push, the *MCC* cannot stop the *FE* chips from transmitting data.

In RunMode there are two ways for the *MCC* to interact with the data flow generated by the *FE* chips:

1. Buffer input data into FIFO’s

   The FIFO is necessary to de randomize the hit data sent by the *FE* and to reduce the data lost due to random fluctuations of the data rate. Since it could always happen that such *FIFO*’s overflow, a Warning mechanism is introduced in the *MCC* EventBuilder to flag truncated events or to re-synchronize the system when needed.

2. Block the LV1 trigger to the *FE*’s.

   This happens when more than 16 events are still inside the module: either in the *FE* chips or in the *MCC* EventBuilder. The *MCC* keeps track on how many LV1 were transmitted to the *FE* chips. In case a Trigger command arrives to the *MCC* before it has finished to process at least one complete event, the *MCC* blocks the LV1 signal to all the *FE*’s it controls. As soon as one event has been fully processed and sent to the *ROD* Trigger issuing is resumed. This mechanism will cause some events to be lost in some detector modules, but will avoid the risk of mixing hits belonging to different events due to lack of synchronization. Each dropped event by the *MCC* is recorded and the information is propagated to the *ROD* in order to keep up with event synchronization.

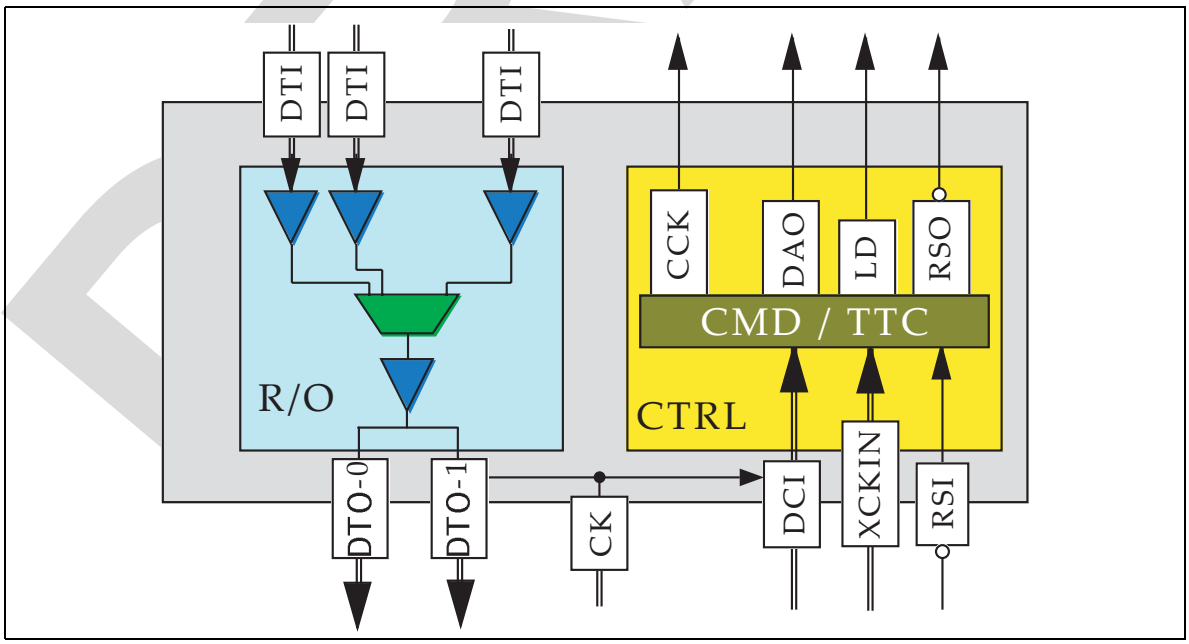

Figure 1-1 *MCC* signal used by the Event Read Out protocol

MCC signals that are active during data acquisition are graphically represented in Figure 1-1:

CK: Input Clock from the *ROD*

CK is a 40 Mhz clock, meant to be synchronous with the LHC bunch crossing. It is transmitted using the LVDS electrical standard. Both signals from DTO and to DCI use this reference clock.

DCI Data and Command Input.

DCI is used to send either data or commands to the MCC. During RunMode DCI is used only for commands. The most frequent command received is Trigger which is encoded using 5 bits, allowing the possibility of generating a trigger every fifth bunch crossing as required by ATLAS. Other commands are the reset or the resync of the *MCC* and *FE* chips.

DTI: Data Input from the *FE*.
Each *FE* chip uses a separate input line (MCC-DTI). The transmission from each *FE* is serial, it uses low voltage differential signaling (LVDS), and data are in phase with the clock XCK. The maximum throughput is 40 Mbit/sec using this data/clock protocol since the clock XCK is 40 MHz.

DTO: Data Output lines to the *ROD*.

The *MCC* puts together different slices of the same event received from the *FE* chips and retransmits with appropriate encoding into the readout chain using the DTO-0 and DTO-1 serial LVDS line. The maximum throughput is 160 Mbit/sec using this data/clock protocol since the clock CK is 40 MHz and the data can be shared on two lines and encoded on both edges of the CK.

LV1: Level 1 trigger.

Trigger signal is generated in response to an 'encoded' trigger signal received by the *MCC* from the DCI input. A Trigger lasts one clock cycle and is synchronous with the XCK clock. Triggers can be suppressed in the case that in the whole module there are more than 16 events to send out.

SYNC: *FE* Synchronize and Reset.

The *MCC* has to provide correct reset signals to the *FE*. There are two synchronization signals that the *MCC* can send to the *FE*'s. One is in response to a Sync command that produces a MCC-SYNC signal that is active for one clock cycle. This command is a Fast command and can be issued without taking the *MCC* out of RunMode.

The second reset command that can be sent to the *FE* is a Slow command and is issued in response to a GlobalResetFE command. It generates a MCC-SYNC signal that does a global reset of all the *FE* chips according to the width of the generated MCC-SYNC signal.

For more information on this commands see Chapter 1.2.2.3, "CAL: Calibration" and Chapter 1.2.3.10, "GlobalResetFE: Reset FE chips".

XCK: Output clock to the *FE* chips and to *MCC* itself.

XCK is generated from the CK clock by the *MCC*. XCK has the same frequency as the CK clock. The XCK signal is routed to all 16 *FE* chips and also to the *MCC* itself that uses this clock as it's main clock. This allows for minimal skew of the clock between all chips on the module.

XCKIN: System clock for the *MCC*.

The XCK output must be externally connected to the XCKIN input. The XCKIN clock is used as master clock for the *MCC*, while the CK clock is only used to synchronize the MCC-DTO and MCC-DTI signals.

1.1 Building blocks

Event building is performed by two concurrent processes running in the *MCC*. The first (Receiver) deals with the filling of the 16 input FIFO's with data received from the corresponding *FE* chips, while the second one (EventBuilder) extracts data from the FIFO's and builds up the events. Each *FE* sends data as soon as they are available with two constraints: event hits must be ordered by event number and for each event an End of Event (EoE) word is always generated. EoE is also sent for the case of an empty event to keep event synchronization.

The block diagram of Figure 1-2 is an expanded view of one of the sixteen FIFO's present in the

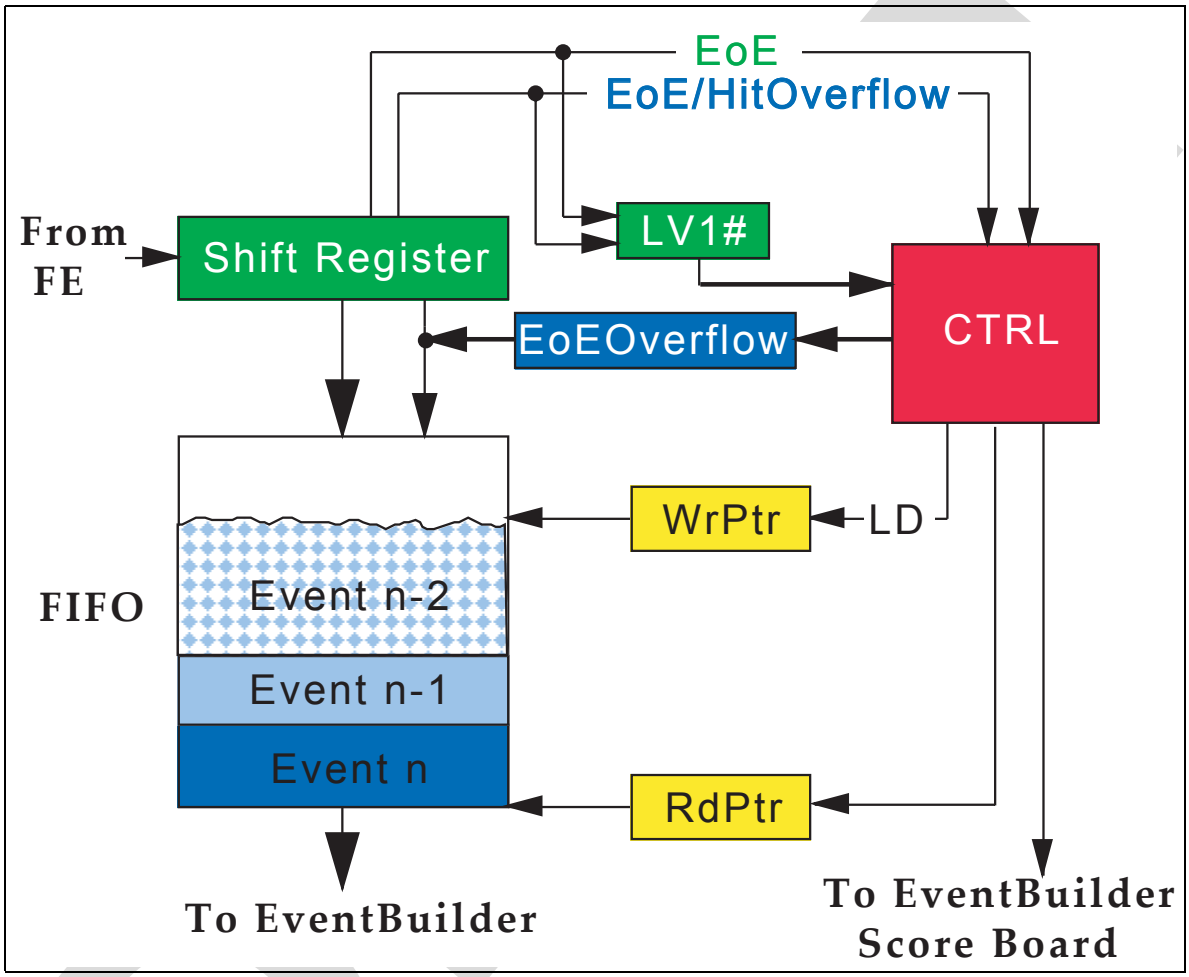

Figure 1-2 Event R/O by *MCC*: Front-End Receiver and input FIFO.

Receiver. As we can see the Receiver is composed by three main building blocks, the receiving Shift Register, the FIFO and the control state machine that deals with Warning conditions and FIFO pointer management.

1.1.1 Input Shift Register

FE's send data serially to the *MCC*. The input shift register is 26 bit wide and continuously reads data from the MCC-DTI line. As soon as the 26-th bit is '1', which means that the *FE* has trans-

mitted a complete Hit or EoE word, the shift register is blocked for one clock cycle and a signal is sent to the control state machine which eventually copies the data in the FIFO.

Data coming from the *FE* while the shift register is blocked is copied to a temporary location in order to allow a continuous data stream.

Another signal that the input shift register provides to the control state machine is if the stored data is a Hit or an EoE word. This allows the control state machine to keep track of how many Hits and EoE words have been written to the FIFO.

1.1.2 FIFO

All data sent to the chip is eventually copied to the FIFO which is used to store data until all *FE* chips have finished transmitting Event data and event building can begin.

The FIFO used in the *MCC* in reality is a 128 x 26 bit wide dual port SRAM full custom block that is used as a FIFO. In order to do this the control state machine provides all the control logic needed for correct pointer management[1-1].

As can be seen in Figure 1-2 the FIFO has two different pointers, one write pointer (WrPtr) and a read pointer (RdPtr).

Both hit and EoE words are stored inside the FIFO and the EoE words are used by the Event-Builder to keep event synchronization.

As described in Chapter 1.3, "FE to MCC Event Format", the Hit word contains Row#, Col# and ToT information while the EoE word contains LV1Id and 4 bit of Warning messages generated by the *FE* chip. If this Warning message is detected the information is written to the corresponding bit of the WFE register inside the register bank (see Chapter 1.1.4, "WFE: Warning from FE").

1.1.3 Control State Machine

This block deals with all signals used to correctly write and read information in the FIFO.

1.1.3.1 Pointer management

Two 7 bit counters keep track of RdPtr and WrPtr. These pointers are incremented each time there is a read or write access to the FIFO. The read access is requested by the OutputPort block of the EventBuilder (see Chapter 1.1.6, "Output Port").

A check is performed in order to avoid pointer overrun or underflow.

As both EoE and Hit words are to be written to the FIFO there are two additional up-down counters that keep track of how many Hits and EoE words have been written. As the *MCC* can accept, like the *FE* chip, up to 16 Trigger commands the EoE counter has an upper limit of 16. The Hit counter is limited to 128 -16 =112 possible values in order to always ensure that we have enough room in the FIFO for writing EoE words.

Figure 1-3 describes a normal readout phase. Each time an EoE word is found in the data stream coming from the *FE* the ScoreBoard inside the EventBuilder is updated. As soon as the Event-Builder starts reading data from a FIFO data is read until an EoE word is found in the FIFO. When this happens the EventBuilder stops reading and starts processing the next receiver FIFO

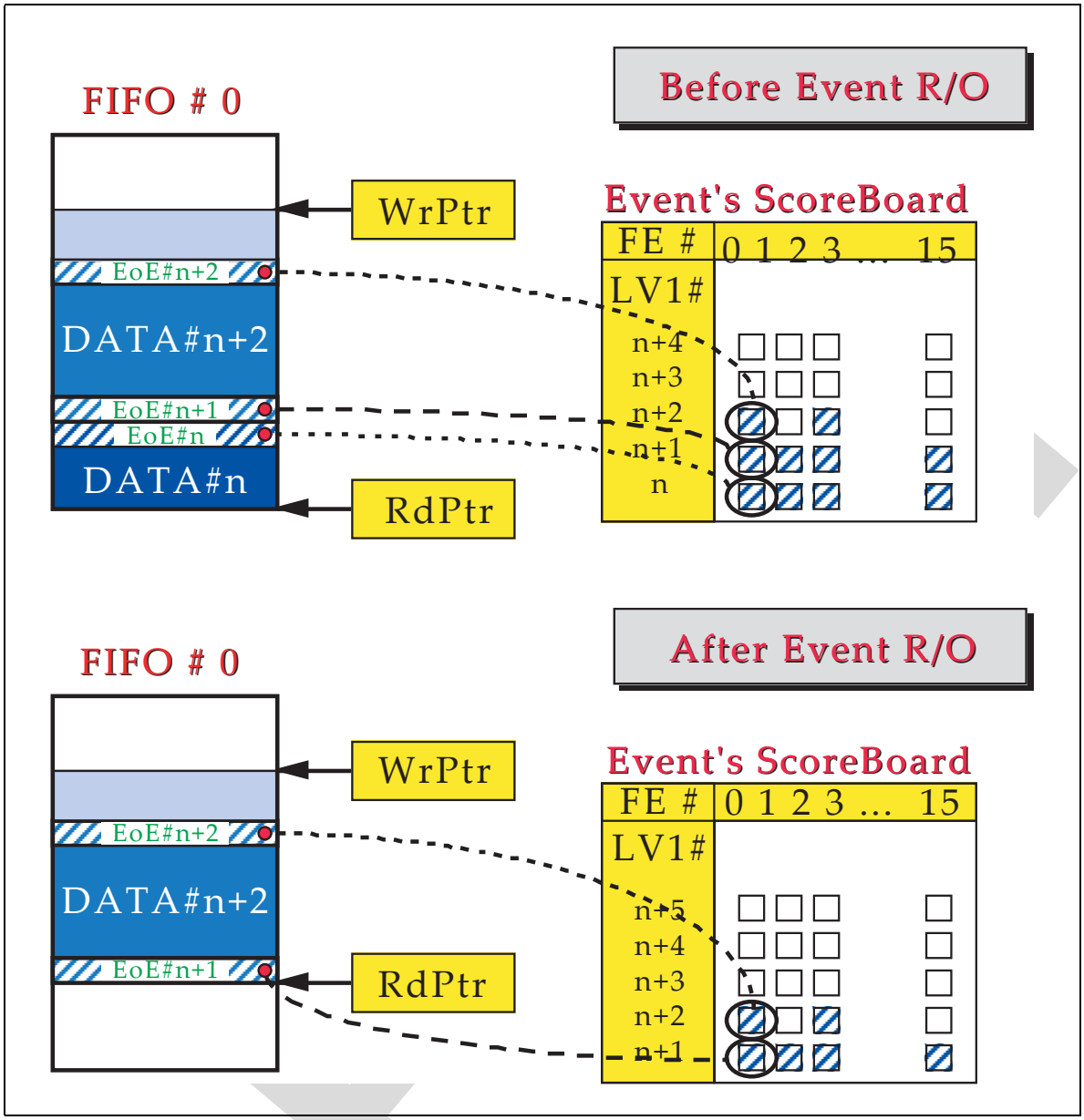

Figure 1-3 Event ReadOut: EoE information is written to the Scoreboard. When event building starts data is read out from the FIFO till the first EoE word is found. Then the EventBuilder proceeds with the next *FE* that contains data until the whole event has been read out. At the same time data can be written to the FIFO.

that contains some data. In case of warning messages inside the EoE word the information is processed by the EventBuilder (see Chapter 1.1.5.1, "FeFlag Generator").

1.1.3.2 Hit overflow

In case of Hit overflow new Hits are simply dropped and in the next EoE word a warning flag is added (HitOverflow) in order to inform the EventBuilder that, in turn, informs the *ROD*. In this case this information is also written to the corresponding bit of the WMCC register (see Chapter 1.1.5, "WMCC: Warning from MCC Receivers").

1.1.3.3 EoE overflow

We also implemented an overflow check for the EoE counter and in this case a different warning flag is added to the EoE word (HitOverflow). Also in this case the information is written to the corresponding bit of the WMCC register (see Chapter 1.1.5, "WMCC: Warning from MCC Receivers").

⚠ In principle this condition should never happen because the *MCC* always ensures that no more than 16 Triggers are sent to the *FE* chips. Being EoE information crucial to correct event synchronization in the whole module we decided to add this feature to be able to check if some data corruption coming from the *FE* chips (for example due to a SEU) has occurred.

1.1.3.4 Data consistency check

Another task that the control state machine has to perform is to check data consistency. This is done checking that all data belonging to the same event has the same LV1Id number. To perform this task the first data word of an event is checked and it's 4 bit LV1Id information (that is present in each *FE* data and EoE word) is copied into a 4 bit register. In all subsequent data words received from the FE belonging to the same event (all words until an EoE is detected) the LV1Id information is compared with the stored one and in case of a mismatch a warning signal (Lv1ChkFail) is issued. The information is written both to the corresponding bit of the WMCC register and to CSR <11>.

1.1.3.5 EoE signals for the ScoreBoard

Three additional control signals for the EventBuilder that this block has to provide are: EoE word detection (TrueEoE), EoE with warning word detection (WngEoE) and IncremenScoreBoard. This task is performed checking, the FIFO word pointed to by the RdPtr, i.e. the next word to be read. If this word is an EoE word without any warning the TrueEoE signal is generated, while if there is a warning (both *MCC* or *FE* warning) the WngEoE is generated. Finally, the IncremenScoreBoard signal is simply the logical OR of TrueEoE and WngEoE.

1.2 Reset Actions

This section will describe the response of the Receiver on all available reset commands that can be used in the MCC.

The only block that is not affected by any reset signal is the full-custom FIFO. For all the others block the reset is synchronous, as throughout the whole chip, and therefore one needs to have the clock active in order to perform a reset of this block.

1.2.1 Response to a Pin Reset

The pin reset (MCC-RSIb) has the same functionality as the GlobalResetMCC command. It is a synchronous reset and therefore one has to ensure that the clock signal is applied in order to be able to reset the chip in this way.

1.2.2 Response to a GlobalResetMCC command

In case of a GlobalResetMCC command the whole Receiver gets cleared.

Also in this case all the counters are reset to '0', the pointers are re initialized and point both to location '0' of the FIFO and the command state machine is put to it's idle state.

1.2.3 Response to a BCR command

The BCR command has no effect on the Receiver.

1.2.4 Response to an ECR command

An ECR command clears all structures in the Receiver with the exception of the FIFO. This means that all the counters are reset to '0', the pointers are re initialized and point both to location '0' of the FIFO and the command state machine is put to it's idle state. This allows for a fast reset of the DataPath of the *MCC* as prescribed by the ECR command.

1.3 References

1-1 K. Kloukinas, "A Configurable Radiation Tolerant Dual-Ported Static RAM macro, designed in a 0.25 µm CMOS technology for applications in the LHC environment.", Proceedings of the "8th Workshop on Electronics for LHC Experiments", 9-13 Sept. 2002, Colmar France